6.2P: Cloud-Native DevOps Project (Part 2 of 5)

Background

Our University Library is a cornerstone of academic resources, aims to enhance accessibility to educational materials through an advanced online platform. You has been asked to develop a cloud-native microservices architecture to support the library's diverse user base and streamline deployment processes. This project is divided into 5 parts as follows:

No.	Title	Task
1	Deploying Microservice with PostgreSQL on Render	Task 4.2P
2	Containerizing Microservices with Docker and Deploying to Local Kubernetes	Task 6.2P
3	Containerizing Microservices with Docker and Deploying to Azure Managed Kubernetes	Task 7.2P
4	Infrastructure as Code with Terraform	Task 8.2C
5	CI/CD with Github Actions using Terraform	Task 9.2D

By end of this project, you will gain a comprehensive understanding of essential DevOps practices and cloud-native application deployment techniques. More specifically, you will be able to do following:

1. Create Dockerfiles to containerize application and define the runtime environment.
2. Develop Kubernetes YAML files (deployment.yaml and service.yaml) to deploy and manage their microservice on Azure managed Kubernetes cluster.
3. Write Terraform scripts (main.tf, variables.tf, outputs.tf, provider.tf) to provision Azure infrastructure.
4. Deploy Azure Kubernetes Service (AKS) and integrate PostgreSQL for data storage.
5. Configure GitHub Actions workflows to automate the CI/CD pipeline.
6. Apply theoretical knowledge to real-world scenarios, enhancing their understanding of cloud computing and DevOps principles.

Tasks

In this task, you are continuing work from the previous project, focusing on Dockerizing the `book_catalog` microservice and deploying it to Local Kubernetes. This task builds upon previous efforts, emphasizing practical skills in containerization, database integration with PostgreSQL, and deployment orchestration with Kubernetes in a local development environment. By the end of this project, you will gain hands-on experience in setting up and managing containerized applications, leveraging Kubernetes for scalability and management efficiencies.

Steps

1. Downlaod code from the **Task Resources** and unzip.
2. Create a github repository named : `<your-name-sit722-part2>`.
3. Add given code to a github repository (local first then push to the remote repo).
4. Write Dockerfile.
5. Create `deployment.yaml` for each microservice.
6. Deploy Microservice to Local Kubernetes.

What will you submit?

You are asked to submit to OnTrack a single PDF document that contains (14 screenshots, URL for your Github Repo, 1 text answer and Explaination of `Dockerfile` and `deployment.yaml`) the following:

1. A screenshot of Render Dashboard showing PostgreSQL resource created.
2. A screenshot of Dockerfile and explanation of each line.
3. A screenshot of the **console** with the output from the commands:
    - `docker build ...`
4. A screenshot of `deployment.yaml` file and explanation of each line.
5. A screenshot of the **console** with the output from the commands:
    - `kubectl config current-context` (Output of this command will be docker-desktop if it not selected as current-context then you have to change it)
    - `kubectl apply ...`
    - `kubectl get pods`
    - `kubectl get deployments`

    - `kubectl get services`

6. A screenshot of Deployed microservices running via Local Kubernetes (`localhost:<PORT>/docs`).
7. A screenshot of adding data of your favorite book. [Add **five** (5) books data but provide screenshot of one data].
8. A screenshot of getting all **five** (5) books data (`localhost:<PORT>/books/`).
9. A screneshot of updating data for any **one** (1) book.
10. A screenshot of the get using **id** of updated book.
11. A screenshot of deleting any **one** (1) book.
12. A screenshot of getting all books data. To verify deleted book data.
13. A screenshot of the **console** with the output from the commands:

    - `kubectl delete ...`

14. A screenshot of Render Dashboard after deleting PostgreSQL service showing empty dashboard.
15. A link to your public github repository.
16. Appendices - provide complete step-by-step instructions for how to deploy application on **Kubernetes**.

Note: Consider providing seperate screenshot for each question (where screenshot is required) and do not crop the same. Provide complete screen.

Complete the Task Page Limit: No page limit for appendices, formatted reasonably, e.g., 2cm margins, 11 or 12 point font, appropriate headings/spacing, etc.